RGB color space cube, is then determined from the equation below:

$$RGB_{anchor} = \underset{RGB}{\operatorname{argmax}} \|RGB - RGB_{ref}\|$$

RGB_{anchor} is one of the vertices of the RGB color cube. A schematic representation of these color points in RGB color space, as well as the line segment extending from RGB_{anchor} to RGB_{ref} (referred to as the "reference axis") is illustrated in FIG. **3**.

The RGB color space is then projected onto the 1-dimensional reference axis with a projection function P( ) such that, $P(RGB_{anchor})=0$ and $P(RGB_{ref})=1$. If any corner projects to a number>$T_{project}$, where $0<T_{project}<1$, then the corner is moved parallel to the reference axis until the projection is equal to $T_{project}$, which is typically equal to about 0.75. As a result of this operation, the 3-D color space is mapped to a 1-D color line segment with RGB_{anchor} and RGB_{ref} at the ends. Reducing the dimensionality increases the speed of the segmentation algorithm. Also, since RGB_{ref} is relatively bright for date fields and relatively dark for scratches or the like, any color space distortion (which occurs if $T_{project}>1$) is not too severe. The above reference and anchor point determination methods, as well as the projection operation, may alternatively be performed in other color spaces such as CIE Lab, etc.

(ii) Next, the reference axis is quantized into K (typically K=256) bins, as shown in FIG. **4**. Each color in the image is now identified by its bin-index (which ranges from 0 to 255 for K=256).

(iii) A "co-occurrence" matrix M[K][K] is created. M[i][j]=the number of pixel locations, such that the current pixel has bin-index i and its right or bottom neighbor has bin-index j. In matrix M, pixels that cluster around the M[i][i] line, shown in FIG. **5**, belong to connected regions with similar colors.

(iv) A threshold t is selected, where 0<t<K, that creates two regions, A and B, in M, as shown in FIG. **6**. The segmenting algorithm considers each possible t and selects t, such that a distinct cluster of pixel data, roughly corresponding to a continuous region, is found in each of A and B. The algorithm uses a "flatness" or "maximum entropy" criteria such that t is selected to maximize the entropy of the data in each of A and B. Mathematically, this process is described as follows.

Let,

$$p_{ij}^r = \frac{M_{ij}}{\sum_{p,q\in r} M_{pq}},$$

$\forall i,j\epsilon r$, where r denotes region A or B.

Choose t such that

$$t = \underset{t}{\operatorname{argmax}} \sum_{r=A,B}\left(-N_r\sum_{i,j\in r} p_{ij}^r \ln p_{ij}^r\right)$$

where N_r=number of pixels in region r.

All pixels with bin-index>t are marked as "missing data."

Alternatively, the segmentation step may be performed by having the user specify a radius, R*. In this case, all colors, such that $\|RGB-RGB_{ref}\|<R^*$ are classified as "missing data."

The segmentation step can be performed for just one sub-portion of the representation or can be performed individually for any number of sub-portions of the image.

Component Filtering

Most of the time regions identified as missing data regions are in fact just that. However, occasionally a region is identified as a missing data region when it is not. One way to handle this situation is to simply treat the region as a missing data region and let the CRFB filtering process estimate the colors, which would not likely have much effect on the quality of the processed image.

However, another preferred way is to perform component filtering to identify any region that has been misidentified as a missing data region. To do this, the area and perimeter length of each missing data region identified in the segmenting step is computed. Any region whose area/perimeter ratio is greater than or equal to a predetermined threshold is discarded from a "missing data region" list. Only regions having an area/perimeter ratio less than the threshold are retained for further processing. The threshold may be set at 0.94, for example, or it may be set by the user. A user may adjust the threshold using an on-screen a slider, for example.

CRBF Filtering

Each missing data region which is retained is now subjected to closest-to-radial-based-function (CRFB) filtering to estimate colors from neighboring pixels to be used to fill in that region. RBF is a function $\phi(x_1,x_2)$ such that $\phi(x_1, x_2)=\phi(\|x_1-x_2\|)$. The relationship between ϕ and $\|x_1-x_2\|$ may be a linear profile or a gaussian profile, as shown in FIGS. **7**(*a*) and **7**(*b*) respectively. One advantage of the CRBF filter is that it combines both spatial distribution and color distribution information about a pixel, Nbd. Moreover, with the CRBF approach, no new colors are introduced in the image. The process is as follows:

(i) A "distance layer" map for each missing data region is created, as shown in FIG. **8**. Each pixel in the missing data region is assigned a layer number which represents the Manhattan-distance to the closest non-missing-data pixel.

(ii) For each pixel p_{ij} in layer L, for each color channel, let, $Nbd_{ij}=\{p_{rs}$: layer_number$(p_{rs})<L$ and $|i-r|+|j-s|<D\}$, where D is typically 1 or 2. For each r, $s\in Nbd_{ij}$, an RBF, ϕ_{rs}, is associated therewith.

Estimate λ_{rs} such that

$$\lambda_{rs}^* = \underset{\lambda_{rs}}{\operatorname{argmin}} \sum_{m,n\in Nbd_{ij}}\left[\sum_{r,s\in Nbd_{ij}} \lambda_{rs}\phi_{rs}(\|(r,s)-(m,n)\|)-p_{mn}\right]^2$$

(iii) p_{ij} is estimated by

$$\hat{p}_{ij} = \sum_{r,s\in Nbd_{ij}} \lambda_{rs}^*\phi_{rs}(\|(r,s)-(i,j)\|)$$

(iv) The pixel value for the i,j pixel is chosen as

$$p_{ij}^* = \underset{p_{mn}\in Nbd_{ij}}{\operatorname{argmin}} \|p_{mn}-\hat{p}_{ij}\|$$

(closest pixel to RBF estimate)

Such an RBF filter advantageously combines both spatial distribution and color distribution information about a pixel nbd. Using a CRBF approach, no new colors are introduced in the image.

An alternative approach for choosing the pixel value for the i,j pixel in step (iv) of the CRBF filtering process is to use $\hat{p}_{ij}$ as the estimate instead of p^*_{ij}. However, the processed image tends to be more blurred in the missing data regions.

Effects and Implementations